A Mediated Definite Delegation Model allowing for Certified Grid Job Submission

Steffen Schreiner[1,2], **Latchezar Betev**[1], **Costin Grigoras**[1], **Maarten Litmaath**[1]

[1] European Organization for Nuclear Research CERN, Geneva, Switzerland
[2] Center for Advanced Security Research Darmstadt - CASED
and Technische Universität Darmstadt, Germany

E-mail: steffen.schreiner@cern.ch

Abstract. Grid computing infrastructures need to provide traceability and accounting of their users' activity and protection against misuse and privilege escalation. A central aspect of multi-user Grid job environments is the necessary delegation of privileges in the course of a job submission. With respect to these generic requirements this document describes an improved handling of multi-user Grid jobs in the ALICE ("A Large Ion Collider Experiment") Grid Services.
A security analysis of the ALICE Grid job model is presented with derived security objectives, followed by a discussion of existing approaches of unrestricted delegation based on X.509 proxy certificates and the Grid middleware gLExec. Unrestricted delegation has severe security consequences and limitations, most importantly allowing for identity theft and forgery of delegated assignments. These limitations are discussed and formulated, both in general and with respect to an adoption in line with multi-user Grid jobs. Based on the architecture of the ALICE Grid Services, a new general model of mediated definite delegation is developed and formulated, allowing a broker to assign context-sensitive user privileges to agents. The model provides strong accountability and long-term traceability. A prototype implementation allowing for certified Grid jobs is presented including a potential interaction with gLExec. The achieved improvements regarding system security, malicious job exploitation, identity protection, and accountability are emphasized, followed by a discussion of non-repudiation in the face of malicious Grid jobs.

1. Introduction

Global eScience Grid infrastructures provide researchers with unified access to computing and storage services across national borders, jurisdictions, and domains of responsibility. Accordingly and beyond the existence of operational and usage policies, accountability needs to be ensured for actions occurring due to the operation of such infrastructures and in the course of its users' activities. Protection against misuse and privilege escalation needs to be established and any violations need to be traceable. These baseline security concerns form the general background and motivation of the work presented in this document.
The ALICE ("A Large Ion Collider Experiment") Grid Services [1], a globally distributed Storage and Computation Grid, are developed and operated by the ALICE Collaboration [2] as a research cyberinfrastructure. Its central Workload Management System (WMS) and File Catalogue are provided by the open source Grid framework

AliEn("ALICE Environment") [3, 4]. The system provides the infrastructure for simulation, reconstruction and analysis of physics data collected by the ALICE detector at CERN, one of the four large experiments within the Large Hadron Collider (LHC). As such, it is embedded within the Worldwide LHC Computing Grid (WLCG) [5], a tiered infrastructure of Grid services for the large LHC experiments. The ALICE Grid Services constitute a Virtual Organization (VO) and are based on 75 computing centres (hereafter referred to as *Sites*) located in 33 countries, combining up to 35k CPU cores and 500PB of storage, and serving approximately 1000 users within the collaboration.

The File Catalogue establishes a central logical layer on top of a globally distributed set of storage servers provided by the Sites, which constitutes one virtual Grid File System for ALICE. The Computation Grid is based on Worker Nodes (WNs) aggregated on Sites within batch farms, receiving Grid jobs from the central WMS. The jobs are specified and represented by a textual description called *JDL* (originating from Job Description Language), listing e.g. the job ID reference number, the file to be executed, execution arguments, and input and output files. Upon submission, Grid jobs are placed into a central task queue as part of the WMS, from which they are fetched for execution depending on order of priority and dependency matching. WMS and File Catalogue form together the so-called *Central Services*.

Fundamentally, the ALICE Grid Services provide two functionality or use cases: the Grid File System access and the dispatch of Grid jobs. Both are surrounded by a rich set of corresponding management and maintenance functionality. The general program code executed within Grid jobs is composed of centrally provided software packages, most importantly the AliRoot [6] software framework, and user supplied code. Thereby, users are free to upload program code and data of any kind into the Grid File System and request it to be executed in Grid jobs. They are legally obliged though, to only utilize the ALICE Grid Services for their research within the ALICE experiment and as such use Grid jobs in order to analyse the experiment's data.

The freedom of the system as a globally distributed cyberinfrastructure creates a challenge for security and in particular of accountability and liability. This document presents an in-depth security analysis of the submission model in the ALICE Grid Services and discusses objectionable and even severe security problems. In the course of the assessment of the applicability of gLExec and Multi-User Pilot Jobs in ALICE we identified the concept of unrestricted delegation based on X.509 proxy certificates as a mechanism for Grid user credentials to be deficient and the its adoption highly questionable. In particular, we found PCs ineligible regarding *accountability*, *non-repudiation*, and due to the potential occurrence of *identity theft*. This document presents the results of this analysis and proposes an alternative approach of *certified* Grid jobs.

In the remainder of this introduction, access control and accountability (section 1) and the implementation of the Pilot Job concept (section 1.2) within the ALICE Grid Services are described. Chapter **??** presents a security analysis of Single-User Pilot Jobs in the ALICE Grid Services and objectives derived therefrom. Within chapter 3, the concept of Multi-User Pilot Jobs with gLExec based on proxy credentials is described, general limitations of unrestricted delegation with proxy credentials are assessed (section 3.1), and the necessary propagation of of these credentials is examined (section 3.2). This is followed by an analysis and specification of the resulting security problems of proxy credentials based on unrestricted delegation with respect to Multi-User Pilot Jobs (section 3.3). A new model of delegation is presented throughout chapter 4 and an implementation of the model presented, allowing for certified Grid jobs (section 4.1) and a potential interaction with gLExec (section 4.2). The concept of non-repudiation in the face of an occurred security incident within a certified Grid job is discussed (section 4.3), followed by a review of the remaining objective of integrity of a Grid job's environment on a WN. Finally, the new model and its implementation are reconsidered with respect to related work (section 5).

1.1. User access in the ALICE Grid Services

Throughout the WLCG, X.509 certificates [7] are used as the basic mechanism for authentication and authorization of Grid users and operators, through middleware provided e.g. by the Globus Toolkit [8], in which the certificates are signed by WLCG recognized Certificate Authorities. Such a user certificate is hereafter referred to as a *Grid Certificate*, while the term *Grid Credential* is used in order to denote the pair of a user's Grid Certificate and the corresponding private key.

Based on the concept of X.509 proxy certificates [9, 10], derived Proxy Credentials (PCs) are further used in order to allow for delegation and single sign-on. Such PCs consist of a generated private/public key pair and a Grid Certificate, whereat the public key is signed with the private key corresponding to the Grid Certificate.

In the ALICE Grid Services, these user PCs are used only upon client logon for authentication and authorization. After a PC with its Grid Certificate is validated in order to grant access, the Grid Certificate's Subject entry is mapped to an ALICE-internal user name based on a LDAP service (see figure 1). Throughout the whole system, Grid users are only represented by their ALICE-internal user name and there is no actual mechanism for delegation of privileges embedded in the system. Whenever a Grid job is submitted by a client, the corresponding ALICE user name is associated to the JDL within the Central Services. Grid jobs are executed on WNs of Sites on behalf of the ALICE VO and user accountability is established based on the internal user name.

1.2. Single-User Pilot Jobs and the AliEn JobAgent

Pilot jobs (PJs) in Grid environments implement an approach to optimize resource utilization and in particular to set up and assess the local environment before a Grid job execution on a WN. The basic principle is to let a WN start running a Grid service instead of an actual job or payload and to let this service handle the execution of one or more actual jobs. Due to the concept of PJs, the actual underlying WN and the batch system in which it is embedded can be fully abstracted and a virtual layer is built on top of all Sites and their batch systems.

In case of the ALICE Grid Services, a Computing Element (a Site service functioning as a resource broker for its batch system) advertises idle resources to the Central Services. In line with these advertisements, Sites are requested to start *JobAgents* (JAs), the Pilot Job implementation of AliEn, according to the load in the central task queue and the Sites' capabilities (see figure 1, red arrows I - V). Once these JAs have started, they will call the Central Services and request actual Grid jobs. Thereby, for each Site the submission of JA requests follows the overall queue status and each Site will receive a request for each matching job in the queue. As a consequence, JAs compete for jobs and the execution of jobs will be assigned as fast as possible with respect to the overall state of the Grid. The JA prepares a job's environment by installing the ALICE software specified in the job's JDL and not yet present, subsequently advertises the corresponding capabilities and can execute a number of consecutive jobs within its own specified lifetime.

The JA uses a PC in order to authenticate and authorize itself to the Central Services (see figure 1). This PC (hereafter referred to as *Pilot PC*) is submitted to a WN by the Site's Computing Element in combination with the JA request, and is based on the Grid certificate of a Site or Grid operator. Once placed on a Site's Computing Element, it is renewed automatically by periodical requests to a MyProxy [11] service. MyProxy is a credential management service that holds long-term uploaded PCs and provides authorized entities with derived PCs of lower order on demand [12]. In a typical usage scenario, a delegator uploads a long-term PC, having a validity e.g. of one month. A Grid service identifying itself with an explicitly authorized Grid credential then can request short-term delegated PCs, e.g. for 24 hours, from the MyProxy service. The delegator can refresh the long-term PC as needed.

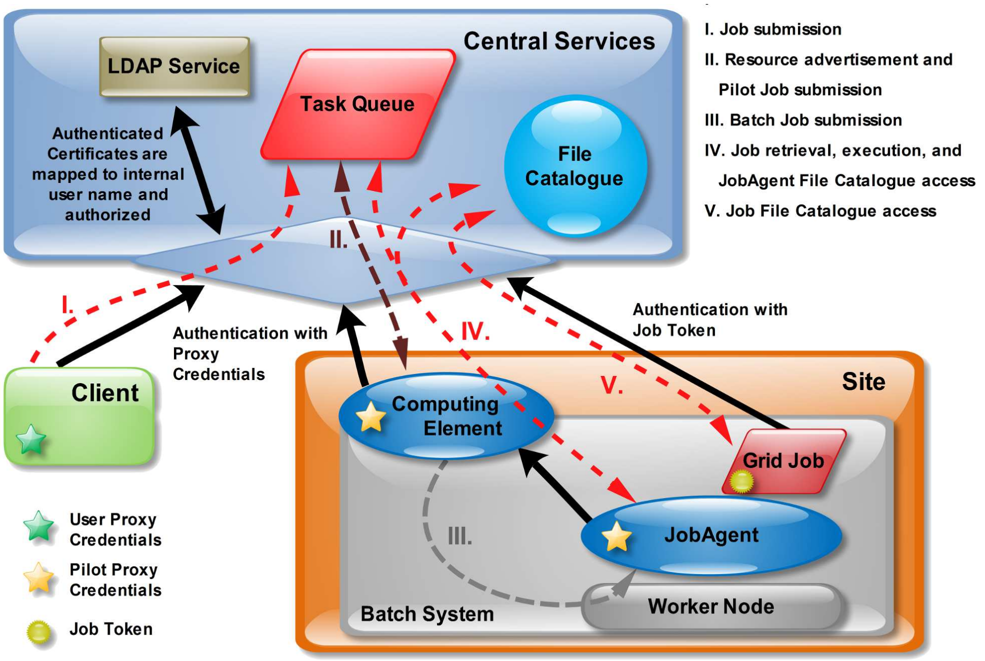
Central Services
LDAP Service
Task Queue
File Catalogue
Authenticated Certificates are mapped to internal user name and authorized
II.
I.
Authentication with Proxy Credentials
Client
IV.
V.
Authentication with Job Token
Site
Computing Element
Grid Job
JobAgent
III.
Worker Node
Batch System
User Proxy Credentials
Pilot Proxy Credentials
Job Token
I. Job submission
II. Resource advertisement and Pilot Job submission
III. Batch Job submission
IV. Job retrieval, execution, and JobAgent File Catalogue access
V. Job File Catalogue access

Figure 1. Single-user Pilot Jobs in the ALICE Grid Services

As the JA executes Grid jobs directly, all jobs are executed on WNs using the same local user account as their respective JA and Grid job user accounting is based only on a job submitter's internal user name in ALICEWith respect to a Site, both the JA and all jobs are executed by the person identified by the Pilot PC, respectively the Site or Grid operator. A JA's Pilot PC has escalated privileges to impersonate a user that submitted a given job, and to retrieve input files from and upload output files to the Grid File System in their name. The job itself can utilize a security token in order to access the Grid file system. It is created on request in the Central Services during the initialization of the job on a WN and revoked once the job finishes.

A JA does not affect a job's execution beyond monitoring and potential termination due to expiration or resource limit exceedance. Both the PJ and its jobs are executed conforming to a Platform as a Service (PaaS) model. The operating system and the JA represent the environment of the Grid job. While the administrative sovereignty of the WN resides at the Site, the sovereignty of the JA program code is at the VO.

> **Definition 1.1**: The approach of multi-user Grid jobs executed by Pilot Jobs which run with single VO-specific users on a WN is referred to as a model of *Single-User Pilot Jobs* (SUPJ).

In contrast to Single-User Pilot Jobs, we define an according multi-user model as follows:

> **Definition 1.2**: The Execution of Grid jobs from different users in a Pilot-Job-based Grid infrastructure with corresponding different local user accounts on a WN is referred to as a model of *Multi-User Pilot Jobs* (MUPJ).

2. Security Analysis of Single-User Pilot Jobs in ALICE

For the analysis of the accountability of Grid users as individuals, the ALICE VO is considered to act as a service provider for its users, and as an intermediary between its users and the Sites as service and platform providers. As this intermediary, it is known as a central broker entity to all Sites, and decides which Grid jobs are supposed to be executed by which Site, and consecutively submits the jobs to the Site provided WNs. Accordingly, the VO receives *task assignments* from its users, while propagating these assignments as Grid jobs to the Sites. Along with the assignment, a Grid user performs an implicit *delegation* of a corresponding subset of its privileges to the WN and the corresponding Site. The Grid Services in combination with the AliRoot framework as a software package provide a Software as a Service (SaaS) to its users, which is based on the PaaS level usage of the Sites' resources. Aside, the Grid Services provide a PaaS to the user's, as any code can be supplied and requested to be executed as a Grid job. The program code and data executed on a WN along with a Grid job in the ALICE Grid Services can be classified according to three internal and one external origins, as specified in table 1.

The model of Single-User Pilot Jobs, as implemented within the ALICE Grid Services, has drastic limitations to security and user accountability. In the following paragraphs, these limitations are discussed and security objectives are defined in order to allow for an analysis of potential solutions. Due to the current architecture of job submission and execution, it is virtually impossible to state the origin of potential security incidents, attacks, or misbehaviour arising along or from a Grid job executed on a WN. Once a user has submitted a job to the ALICE Grid Services, the job is completely in the sovereignty of the VO. As the relation of a Grid user and a job is provided by the internal user name only, this relation is fully controlled within the Central Services. Similarly, the VO has the control to deliberately alter a user's job submission. Within the Central Services and before their submission to Sites, the original job requests are processed. In the course of this processing, jobs can be split into sub-jobs working on only subsets of the specified input data. Once at a Site, there is no further insurance of a correct execution of a job beyond simple run time and resource utilization monitoring by the Pilot Job. Finally, as a consequence, a Grid user has no possibility to

Internal I: Detector and user data, and user code.	Data stored in the Grid File System within files. This can be the data of the ALICE detector, both raw or preprocessed through several stages, or any data or program code supplied by user's of the system.
Internal II: Software Packages	Program code downloaded and supplied to the job by the JobAgent, provided by the VO as software packages.
Internal III: Worker Node	Program code provided within the operating system of the WN, such as system commands and libraries.
External	Data retrieved within the job directly from external or third party resources, e.g. via downloads from the Internet.

Table 1. Data origins in the ALICE Grid Services

prove non-conformity of the actions taking place in their name and as such to protect the misuse of its identity, which results in the following security objectives:

Objective 1, *Provable authenticity of assignment*: The original submission of a Grid job must be verifiable at any later stage, proving the submitter's identity and the assignment as it was submitted.

Objective 2, *Provable authenticity of assignment processing*: The processing of a Grid job as an assignment must be verifiable at any later stage, proving to result from a set of sound transformations performed within the Central Services, respectively by the VO only.

Objective 3, *Protection against forgery of assignment*: Grid job assignments must not be forgeable by Grid users, the VO, the Sites, or any third party.

Objective 4, *Protection against misuse of delegation*: The delegation of privileges along with a Grid user's job submission must be protected from being misused.

The objectives 1 to 4 represent necessary criteria for non-repudiation of both a Grid user's job submission and the job's processing within the VO's control.
According to the SUPJ model, the JA on a WN executes all Grid jobs as received on behalf of the ALICE VO, in which the jobs are executed directly by the JA process and therefore run within the same local user environment on the WN. Hence, Grid jobs are not strictly isolated from each other within their execution. A JA runs only one Job at a time and a job's working directory is scratched after the execution. Nevertheless, a job can fork sub-processes that will remain after its execution on the system, running with the same user account and privileges, and are therefore for example able to alter later executed jobs. Further, jobs are neither encapsulated nor isolated with respect to their JobAgent, and are therefore able to alter the JobAgent or get hold of the

Pilot PC. This introduces a crucial security issue to the Grid. A Grid job could start e.g. the AliEnGrid client on the WN, while utilizing the Pilot PC, and submit new jobs. This would enable an attacker to conduct for instance denial of service attacks. Further on, the Pilot PC entitles to act on any user's behalf within the ALICE Grid, as this is needed in order to handle file uploads in the name of the job and thereby in the name of the job submitter. As a consequence, any holder of a Pilot PC is currently able to impersonate any ALICE user. Accordingly, we further formulate the following security objectives:

> **Objective 5**, *Grid job isolation*: Grid jobs should be mutually isolated and must be prevented from potential mutual interference, both concurrently and consecutively in time.
>
> **Objective 6**, *Pilot Job protection*: A Pilot Job must be protected from alteration, interference, or disruption by one of the Grid jobs it is executing. Analogously, its Pilot Job credentials need to be protected from any misuse by jobs.
>
> **Objective 7**, *Pilot credential limitation*: Pilot Job credentials must be limited in power, not to allow any escalated privileges, in particular with regard to Grid user's identity, in order to impede misusage by Grid users, the VO, the Site, or third parties must be protected from any misuse by a Grid user, the VO, a Site, or thirds.
>
> **Objective 8**, *Pilot platform integrity*: The WN and its Pilot Jobs as the Grid job platform, must provide an environment of integrity and be secured of any non-conform Site access or access of any third party.

Grid jobs originating from different users are not visible to a WN's operating system, and by that to the Sites, in a transparent way and it is not possible to enforce a per-user Grid job control. In case of security incidents or attacks originating from single Grid user accounts, it is not possible to respond accordingly and potential counter-measures can only affect a VO's entire set of jobs on a Site or WN. Finally, the architecture prepares no possibility for Site-based usage and resource accounting on a per-user level. Accordingly, we state the last further objective:

> **Objective 9**, *On-Site Grid job user accounting*: Grid jobs need to be authenticated and authorized in a transparent way in the operating system of a WN.

3. Multi-User Pilot Jobs with gLExec and Proxy Credentials

In order to allow for a secure handling of Multi-User Pilot Jobs, the Grid middleware gLExec [13] was developed. Instead of a direct execution of a job or payload, gLExec can be invoked by a PJ, in order to enable authentication and authorization of an associated identity and to allow for isolation of a job's or payload's process. The authentication and authorization is based on the proxy credential of a Grid user, hereafter referred to as *Grid user Proxy Credential* (GuPC), provided to gLExec through an environment variable. Isolation of a Grid job, and its separation from the PJ, can be obtained through a user and environment switch within a POSIX-compatible operating system, similar to the UNIX `sudo` command. Depending on its configuration, gLExec would map a given GuPC to a local POSIX user ID whose value would usually be different for distinct values of the Grid Certificate's Subject line. It is also possible for a site to configure gLExec without such an identity change, whereby the user job or payload will run with the same local user ID as the PJ itself (assuming the GuPC was authorized). Within this document, we will consider the utilization of gLExec only for the case of a full invocation with enforced authentication and authorization and a subsequent certificate-mapping-based user switch. Moreover, we assume this operating mode to be able to comply with objectives 5 and 6 defined above. This assumption implies a sound setup and invocation of gLExec and lies within the limitations of the achievable degree of isolation of different user accounts inside the same POSIX-compatible operating system.

3.1. Security limitations of unrestricted Proxy Credentials

The X.509 PCs as they are in use throughout the WLCG and beyond are based on unrestricted delegation. As such they have long-known cardinal security limitations [9], which were already considered while X.509 PCs were being adopted as a functionality [14]. In this section, we specify and describe from a conceptual perspective four essential limitations of PCs with respect to their usage in the WLCG (while disregarding auxiliary restrictions that can be applied beyond):
Unrestricted delegation based on proxy signatures can be illustrated by a mathematical representation of the delegation as a function. Let $U = \{u : \text{user able to delegate privileges}\}$, $P_u = \{p_u : \text{delegable privilege of a user } u \in U\}$, and $T = \{t : \text{time stamp in seconds}\}$. With $t_{issued}, t_{expires} \in T$ as the two time stamps of beginning and end of validity, the delegation of privileges based on the usage of X.509 proxy certificates, can be expressed as the function

$$\gamma_{\text{PC}} : U \times T \to \{ \bigcup_{u \in U} P_u, \emptyset \}, \quad \text{with} \quad \gamma_{\text{PC}}(u,t) = \begin{cases} \emptyset & \text{if } t < t_{\text{issued}} \text{ or } t \geq t_{\text{expires}} \\ P_u & \text{if } t_{\text{issued}} \leq t < t_{\text{expires}} \end{cases} . \tag{f 3.1}$$

As such, the delegation is not only unrestricted, but also has no dependencies other than in the dimension of time.

> **Limitation 1**, *Unconditional delegation*: A PC itself has neither any binding to a particular delegate nor any context-sensitivity of its usage. Due to this, any privilege is held as such and any limitation or binding would require additional external mechanisms in place.

> **Limitation 2**, *Unrestricted delegation*: Except in time, a PC allows only for an unrestricted delegation to the delegate and thereby holds all privileges of the delegator as such.

A PC within the WLCG has a validity of typically hours or days, which cannot be considered too little for a successful exploitation by potential attackers. In scenarios including the use of MyProxy services, a PC can in principle be extended in lifetime by a renewal request to a corresponding MyProxy service, as long as the original first order PC inside the service is still valid. The renewal can be controlled by the use of

keys and usually is based on specific properties of the requester's own PC (typically its Subject), though the effectiveness of these mechanisms is dominated by secrecy of the keys and security of the involved systems.
An extension to the proxy mechanism called VOMS (Virtual Organization Membership Service) [15] provides the possibility to apply authorization attributes to a PC. On presentation of a valid PC to a VOMS server, an entity can request the addition of any attributes to which it is entitled, e.g. VO membership or roles. Security can be improved by application of a VOMS-based authorization setup in which a plain PC without the necessary VOMS extensions would not be sufficient to conduct any tasks in the system. However, within the WLCG environment the VOMS extensions are primarily used to elevate the privileges of a PC holder, e.g. to obtain the necessary role to run a PJ. Without explicit preventions, any holder of a plain PC can request a new set of VOMS extensions within the range of privileges of the original certificate owner.
As such, PCs may be, once obtained and within their validity in time, fully exploited for identity theft and their misuse could lead to severe security consequences. Regarding GuPCs, this discussion can be further substantiated by the WLCG EGEE Grid Security Policy: *"Users [...] must ensure that others cannot use their credentials to masquerade as them or usurp their access rights. Users may be held responsible for all actions taken using their credentials, whether carried out personally or not. No intentional sharing of credentials for Grid purposes is permitted."* [16]

Limitation 3, *Exposure to theft*: A PC is by itself completely unprotected, while being handed on within a distributed system such as a Grid environment. Regarding this aspect it is comparable to a plain security token. Without additional protection a PC can be stolen at any point where it resides and must be expected to be accessible by persons with privileged access.

Limitation 4, *Multiple domain validity*: A Grid Certificate and hence a PC can be recognized by more than one VO or set of service providers, and thus be used to access all the resources concerned.

Limitation 4 would apply when different VOs or sets of service providers accept the same PC for a particular person. Within the WLCG VOs such a situation is very rare: some persons are members of multiple VOs, but usually with a unique Grid certificate Subject per VO (for practical reasons). The limitation would in particular affect users with administrative functions in at least one of their VOs. As a consequence of the limitations 3 and 4, a low level of protection of PCs in one VO could drag down the level of protection in another VO.

3.2. Propagation of Grid user Proxy Credentials

The application of gLExec relies on the propagation and transmission of GuPCs from the actual job submitter and owner of a Grid Certificate to the Pilot on the WN where the job is supposed to be executed. Yet on the part of gLExec, there are no specifications or particular information regarding a potential implementation of this process. We therefore briefly outline two different approaches taken by the LHC experiments *ATLAS* and *LHCb*:
In case of ATLAS [17, 18], the adoption of gLExec is based on the utilization of one

central MyProxy service located at CERN, into which GuPCs are uploaded protected with random keys that are kept within the VO's sovereignty in their WMS implementation. Once a PJ on a WN connects to the WMS in order to retrieve a job, together with the job description it will receive the key, which is then used by the PJ to retrieve a valid user GuPC from the MyProxy service.

> **Definition 3.1**: The approach in which a VO holds keys with a one-to-one relation to the actual GuPCs, although it will not store or transport the GuPC directly, is considered an *indirect GuPC propagation.*

The integration of gLExec into the LHCb [19] WMS follows a different methodology, while not utilizing the MyProxy service for that purpose. The GuPCs are managed directly by the VO's central service and the pilot receives the GuPC together with the payload.

> **Definition 3.2**: The approach of a proprietary storage within the VO's sovereignty and implicit transfer of GuPCs is considered a *direct GuPC propagation.*

3.3. Consequences of Grid user Pilot Credential propagation

Without comprehensive additional mechanisms, the above described limitations of PCs based on unrestricted delegation lead to fundamental weaknesses concerning Grid job user accountability. Their adoption as GuPCs can introduce severe security threats in matters of MUPJ frameworks. Subsequently, these weaknesses and threats are discussed as security problems:

> **Problem 1**, *Unprovable correlation of assignment and delegation*: A job submitter's GuPC on a WN does not have any binding to any actual job or payload. The availability or presence of a valid GuPC is neither a binding statement to prove the authenticity of a job requested to be executed on a WN, nor does it prove to be an authentic derivative of a user's submission.

As a consequence and without further controls, the use of GuPCs is not able to fulfill the requirements for accountability of users with respect to job submissions. GuPCs could be potentially stolen, misused, or mixed up at various points between the user's job submission and a WN without notice. Similarly, a job's description or payload could be altered or exchanged. In [20] this concern was raised as the necessity to trust a VO to provide flawless correlations between GuPCs and submitted jobs.

> **Problem 2**, *Fuzzy validity and expiration*: The validity of a GuPC is by itself independent of the validity or lifetime of a Grid job.

Without explicit functionality, a GuPC must be assumed to be still valid once a corresponding Grid job has terminated. In case of an indirect GuPC propagation using a MyProxy service, the relation between GuPCs and Grid jobs cannot be assumed to be bijective, viz. the same GuPC could (and actually will) be used for several Grid jobs, to reduce the credential management overhead. In any case, a GuPC could potentially be renewed, a GuPC could potentially be renewed until the corresponding first-order GuPC stored in the MyProxy service expires. In principle also the latter credential could even be renewed periodically, which would lengthen the potential validity of all GuPCs derived from it. In the worst case, this could lead to GuPC validities up to many months.

> **Problem 3**, *Unlimited access of VO and Sites*: Even if GuPCs are never stored or processed within a VO WMS, as with the indirect GuPC propagation, a user or attacker holding certain privileges within the WMS must still be considered able to retrieve any active user's GuPC. Since a GuPC must be readable to the PJ process on the WN which manages the handover to gLExec, at least anybody who can control the PJ (within the VO or the Site) would be able to retrieve GuPCs.

> **Problem 4**, *Elevated trust in the VO and Sites*: In comparison to the SUPJ scenario, both direct and indirect GuPC propagation for MUPJs signify an elevated level of necessary trust in the VO and the Sites. While in the SUPJ any user or attacker that has access to certain privileged services can only harm the concerned VO itself and exploit a user's privileges within the VO, the holder of a GuPC might also access a third VO or service that happens to recognize the GuPC. As a consequence, not only do the certificate owners have to put an increased trust in all VOs they are working with, but also the required level of mutual trust between VOs is raised accordingly.

> **Problem 5**, *Challenge of storage*: For both the direct and the indirect GuPC propagation the main storage entity of GuPCs becomes a critical security concern. The storage must be instantiated and maintained secure. Attacks on the storage must be considered severe security threats.

> **Problem 6**, *Drawback of additional service invocation*: A scenario utilizing remote service callbacks, e.g. the indirect GuPC propagation, introduces additional risks of mitigated availability due to failures or attacks. Moreover, any additional invocation amounts to additional dependencies, additional load in matters of scalability and the introduction of delays.

As a consequence, the application of gLExec based on the presented alternatives for the implementation of the propagation of GuPCs amounts to no change or improvement regarding the significance in accountability. In both the SUPJ and the MUPJ scenarios described above, the VO is able to submit jobs to Sites in the name of a user with neither the user's nor the Site's notice. Consequently, the presence of a user GuPC cannot be considered at all as proof or anchor for accountability and an implementation would not be able to fulfill the criteria reflected by the objectives 1 to 4, 7, and 9. A sound implementation of job accountability does not only allow for a user to be proven accountable for a certain malicious or illegal behaviour that was observed, but also ensures a user can rightfully disclaim responsibility when the user's behavior was appropriate. Consequently, the objective must be to allow not only for identification of the user who submitted a job, but also for verification of the actual job at hand. In order to prove the authenticity of a Grid job on a WN, first its origin needs to be provable to be an authentic submission by a particular user and second, any alterations of the job by the VO need to be verifiable.

4. Mediated definite delegation

By analyzing the job submission model in the ALICE Grid Services, we found no reason to justify the utilization of a full delegation approach in order to allow for MUPJ. In contrast, JDL-based job requests describe the required permissions in a context-sensitive expression, and as such implicitly state the least privilege of a necessary delegation.
Facing a three-tier design of service users (Grid users), a service broker and processor (VO), and a service/platform provider in the back end (Sites), the delegation by a user to a Site or WN in the course of a Grid job is mediated by the VO. Aligned to this design, we present a new mediated delegation model based on the delegation of definite privileges, in order to access explicitly specified system entities, to agents assigned by a broker after applying verifiable transformations.
With respect to Grid environments, let $U = \{u : \text{user able to delegate tasks}\}$, $P = \{p : \text{delegable privilege}\}$, $E = \{e : \text{accessible entity, as e.g. files, jobs, or services}\}$, $A = \{a : \text{agent, able to execute tasks on behalf of users}\}$, and $T = \{t : \text{time stamp in seconds}\}$. Further let $C = \{c : \text{concession}\}$, wherein each element describes the delegation of $p \in P$ in the name of $u \in U$ with respect to $e \in E$ to an $a \in A$ at a certain point in time $t \in T$.

Definition 4.1: A delegation based on the transfer of concessions is defined as *delegation of tasks* or *static delegation*, and is expressed by the following mapping

$$\delta : U \times P \times E \times A \times T \to C \ . \tag{f 4.1}$$

Let $B = \{b : \text{task request broker}\}$, $\bar{C} = \{\bar{c} : \text{non-mediated concession}\}$ describing delegations of $p \in P$ in the name of $u \in U$ with respect to $e \in E$ at a certain point in time $t \in T$ to be mediated by a $b \in B$, and $D = \{d : \text{derivative, a verifiable transformation}\}$ describing derivatives a $b \in B$ can apply to a $\bar{c} \in \bar{C}$ upon mediation.

Definition 4.2: $\bar{C}$ is defined by a *mediation request for task delegation* as a mapping

$$\phi : U \times P \times E \times B \times T \to \bar{C} \ . \tag{f 4.2}$$

Definition 4.3: A *mediation of task delegation* is defined as the derivative or transformation of a $\bar{c} \in \bar{C}$ according to a D and the assignment to an $a \in A$ at a certain point in time $t \in T$, with the result being an element of $\hat{C} = \{\hat{c} : \text{mediated concession}\}$. It is expressed by the mapping

$$\psi : \bar{C} \times \mathcal{P}(D) \times A \times T \to \hat{C} \ . \tag{f 4.3}$$

A mediated delegation of tasks or *mediated definite delegation* can then be expressed by a composition of the mappings, as

$$\delta_{\text{mediated}}(u, p, e, b, t', D, a, t'') = \psi(\ \phi(u, p, e, b, t')\ , D, a, t'') \ .$$

The mapping $\delta_{\mathrm{mediated}}$ then describes a definite static delegation with respect to the entities the delegated privileges apply to, but is dynamic with respect to the agent to be elected and using the delegated privileges during task execution.

Within the ALICE Grid Services, the Central Services act as a broker, deciding where a Grid job is to be executed and thus determining which agent receives a Grid job for execution. The JDL of a job is independent of this decision, and any transformations in the course of its processing within the Central Services describe a refinement or derivative of the specified information. As such, it is possible to view the processing as refinements, while only appending information to the original JDL.

The ALICE Grid File System is based on the abstraction of a logical and a physical layer, with a File Catalogue containing logical file entries as references to physical files stored in distributed storage services. By replicating logical files to several storage servers the system is improved with respect to resilience and allows for optimizing data access according to proximity [21]. The File Catalogue access from within Grid jobs is based on this mechanism, and the JDL refers to logical file entries, while allowing the user to specify storage servers as preferences or constraints. The access is granted based on a central ticketing service, which incorporates control over the storage server selection. Accordingly, the physical file access requests in line with a Grid job can be modelled as derivatives of the logical specification within a JDL.

Finally, the model can be extended to allow the propagation of non-mediated concessions between several brokers before their assignment to an agent. Therefore, the set $\bar{C}$ is redefined, so its elements can be assigned not only to an $a \in A$ but as well to a $b \in B$. Then a function ϱ must be defined to allow for a

$$\delta^{\mathrm{n}}_{\mathrm{mediated}} = \psi(\ \varrho^{\mathrm{n}}(\ \phi(\ldots)\ ,\ldots)\ ,\ldots)\ \ ,$$

where n is the number of subsequent applications of the mapping ϱ representing relaying between brokers. The case $n = 0$ represents the mapping $\delta_{\mathrm{mediated}}$ as defined above.

Definition 4.4: A *relaying of mediation request* is defined as the propagation of a $\bar{c} \in \bar{C}$ to another $b \in B$ while applying derivatives or transformations according to a set D at a certain point in time $t \in T$. It is expressed by the mapping

$$\varrho : \bar{C} \times \mathcal{P}(D) \times B \times T \to \bar{C}\ \ . \tag{f 4.4}$$

4.1. Certified Grid jobs

The described model of mediated delegation can be implemented based on composite digital signatures of a JDL in the job request. As user Grid Certificates together with their corresponding private keys can be used for digital signatures, we propose to require a client to sign the JDL upon submission. By utilizing a signed JDL (hereafter referred to as an *sJDL*) throughout the whole propagation of the job, while also providing the corresponding user Grid Certificate, it is possible to prove at any point the integrity of a job and to identify illegal modifications. A signature allows proving the authenticity of the JDL using the user's public key, while the Grid Certificate states a user's membership and authorization within the VO. The user Grid Certificate is appended for both signature verification and certificate validity evaluation. Using time

stamps within the sJDL, set by the user and verified by the VO upon submission, an sJDL allows for a delegation with warrant, stating the user wanted a particular job to be executed as such and within the specified time frame.
In order to further state the approval of a job to a Site, a central service of the VO signs the sJDL as well, viz. as a broker, using an appropriate public-private key pair of a VO-specific Grid Certificate. This second signature is applied before a job is sent to a Site and includes all information appended by the VO and states potential job transitions, e.g. splitting. This doubly signed JDL will hereafter be referred to as *s_2JDL*. On the WN an authenticated JA gets authorized as a delegate of the job it is asked to execute, according to statements in the job's s_2JDL. For (long-term) accountability the s_2JDL needs to be stored within the Central Services. If additionally recorded at the Site, it can be used as proof of the retrieval of a particular job. The proposed protocol for Certified Grid Jobs in ALICE is detailed in the steps below and in figure 2. The Client and Computing Element authentications based on PCs stay unchanged and are omitted.

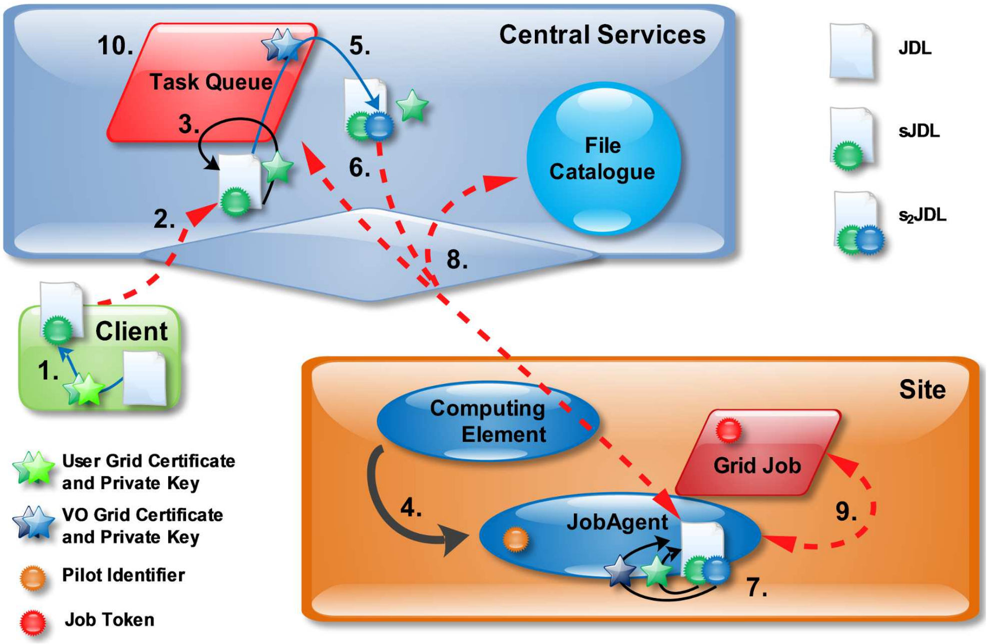

Figure 2. Certified Grid Jobs

Step 1: The client sets a submission and expiration time stamp in the JDL and signs it using the user's private key.

Step 2: This sJDL is sent to the central service of the VO together with the user's Grid Certificate.

Step 3: The signature of the sJDL and the submission and expiration time stamps are validated within the Central Services.

Step 4: Upon request from the Central Services the Computing Element submits a JA as a batch job, with a JA identifier and the certificate identifying the Central

Services.

Step 5: The JA authenticates itself to the Computing Element with its identifier and requests a job from the Central Services through the Computing Element. In case of a matching job waiting in the Task Queue, the Central Services sign the sJDL, including a submission and expiration time stamp and the JA identifier. The use of a second pair of time stamps is necessary as a job could be resubmitted by the VO in case of errors. While the user's submission and expiration time stamps define the time window during which the VO is entitled to send the job to Sites, the VO's time stamps in the second signature can be adapted to the desired run time of a job on a worker node and as such can define a much shorter time window.

Step 6: The resulting s_2JDL is sent together with the corresponding user's Grid Certificate to the JA and the job is marked in the central Task Queue to be taken by this JA.

Step 7: The JA verifies both signatures, based on the authenticity of the signatures, the validity of the time windows of submission and expiration time, and the two certificate chains. In order to record a job's execution for accountability, the s_2JDL can be logged within the Site.

Step 8: Although there are currently no limitations on read access in case of the ALICE Grid services, the s_2JDL could be used for explicit restriction. Once a job has finished or is stopped, the JA uploads the job's specified output files in the name of the job submitter, while receiving only write permissions as stated in the s_2JDL on file or directory level.

Step 9: The Grid File System write access for the job itself can be limited according to the s_2JDL as well, if users are required to specify all output files explicitly in advance. Before a job is executed, the PJ could start a communication server which offers a local service in order to provide Grid File System access for the AliRoot framework within a job. The connection would be secured by a key which is propagated to the job upon its start-up.

Step 10: Once the output files of the job are uploaded by the PJ, the job reaches a final state (e.g. *DONE* or *ERROR*) and the use of the s_2JDL as a token is invalidated in the central services.

If the PJ directly or indirectly manages all external communications, the job itself no longer needs a connection to the Grid services. Thereby, all job-related accesses on the Grid File System can be authenticated by the JA's identifier and authorized based on the s_2JDL. This design directly allows for a future adoption of gLExec.

The composite signature of the JDL represents the realization of the *mediated definite delegation* as formulated above in the δ_{mediated} mapping. The first signature represents the ϕ mapping (f 4.2) with its sJDL outcome describing a series of *non-mediated concessions* as elements of $\bar{C}$. Correspondingly, the second signature represents the ψ mapping (f 4.3), while its s_2JDL outcome describes a series of *mediated concessions* as elements of $\hat{C}$. Furthermore, the implementation prepares for signatures of multiple brokers, according to a potential use case of the relaying of mediation requests (f 4.4).

The mechanism allows for fulfillment of the security objectives 1-4, as both the assignment's and assignment processing's authenticities are provable and their forgery is prevented by the secrecy of the respective private keys. Regarding objective 7, the PJ identifier is only used for authentication and awards no further permissions beyond retrieving s_2JDLs in order to run the corresponding jobs. Therefore, threats like the submission of new Grid jobs exploiting arbitrarily the identity of other users within a job or any other attack or illegal behaviour based on escalated privileges obtained through a Pilot PC are dissolved (see section 4.4). Nevertheless, access to the PJ by a job and mutual job interference cannot be impeded (objectives 5 and 6). Regarding objective 9, *On-site Grid job user accounting*, the described scenario allows for logging of the certificate information of the job submitter, which is provided along with the s_2JDL on the PJ's request for a user job. However, Sites still have to entrust the VO with the authentication and authorization of jobs, since the jobs' verifications would take place inside the VO-supplied PJ code.

The functionality introduced for certified Grid jobs precisely follows the current communication and service invocation schema of the ALICE Grid Services and thereby introduces no additional remote invocations or callbacks between Grid job submission and execution. It further requires no renewal of credentials or delegations. The additional cost in computation due to signature generations and verifications and the necessary storage can be considered negligible. In line with a feasibility study of the mechanism proposed above, a prototype using an encrypted connection between the PJ and the Central Services was implemented. The implementation is based on standard security libraries only, utilizing *SHA384withRSA*-based signatures provided by the BouncyCastleProvider[22] and encrypted communication based on *SSLSocket* from the *javax.net.ssl* package. A sample of an s_2JDL as utilized within the prototype and allowing for serialization is given by listing 1.

```
<SJDL><NOTBEFORE>1312392035</NOTBEFORE><NOTAFTER>1313601635</NOTAFTER><NESTEDJDL>
  <SJDL><NOTBEFORE>1312392035</NOTBEFORE><NOTAFTER>1313601635</NOTAFTER><NESTEDJDL>
    Executable = {"cat"};
    Arguments = {"myInputFile"};
    InputFile = {"/catalogue/data/myInputFile"};
    Output = {"stdout","stderr"};
    User = {"testuser"};
    Broker = {"myVO"};
    HashOrd = "Executable-Arguments-InputFile-Output-User-Broker";
  </NESTEDJDL><SIGNATURE>FTi2ATSgQ[...]CoA0TG==</SIGNATURE></SJDL>
  PilotIdentifier = {"FpK0bE9P[...]Jq1zNx"};
  HashOrd = "SJDL-PilotIdentifier";
</NESTEDJDL><SIGNATURE>EMQlV0Wzg[...]r47ivk=</SIGNATURE></SJDL>
```

Listing 1. A sample s_2JDL (identifier and signatures truncated)

4.2. Certified Grid Jobs and gLExec

Assuming a gLExec modification to allow for authentication and authorization of s_2JDLs (largely an imitation of what the PJ does itself), further fundamental improvements could be achieved: by validating the s_2JDL and the accompanying public user certificate, not only the submission by a certain user can be ascertained (as in a GuPC-based scenario), but actually the submission of the particular job at hand. Presuming the gLExec user-switching mode, *Grid job Isolation* and *Pilot Job protection* (objectives 5 and 6) can be fulfilled completely within the limitations of the operating system's user separation, as well as *Pilot credential protection* (objective 7). Taking this further, Ref. [23] describes a mechanism for trustable meta information on files in the File Catalogue to be provided by the underlying storage systems.

In combination with s_2JDL and gLExec it would then be possible to ensure not only a job's authenticity, but also the authenticity of the files referred to within its s_2JDL. Given the Pilot protection and the mutual job isolation, a job cannot influence executable, arguments, or input files of other jobs running on the same WN, as these are controlled by the protected PJ and based on the sJDL. Assuming *Pilot platform integrity* (objective 8) including a sound gLExec processing, this allows for full *On-site Grid job user accounting* (objective 9): the identities of files added by any user would be recorded by the trusted storage systems in checksums, which are stored in the File Catalogue. A signed Grid job then acknowledges the user's intent for the job to be executed as stated by the sJDL and thereby based on the referred files as they appear in the File Catalogue at the moment of the job's submission. The time stamps in the sJDL, the file alteration time stamps for entries within the File Catalogue, and the checksum verifications during transmissions of the files allow for detailed assessment and accounting of the job before and after run time (see section 4.3).

Our approach of s_2JDL-based job authentication and authorization was presented to the gLExec development team and has been accepted as a potential development in the form of a suitable plug-in.

4.3. Non-repudiation of Certified Grid Jobs with gLExec

In this section we discuss the conditions for accountability and in particular non-repudiation with respect to an attack having arisen from a certified Grid job executed via a modified implementation of gLExec as previously specified. If not stated otherwise, we will consider the Pilot platform to perform with integrity (objective 8), as well as the PJ itself and the VO, and the VO-provided software packages to be non-malicious. We examine exculpatory and inculpatory evidence, while referring to the four different data origins specified in table 1.

In case a hostile Grid job was executed, malicious code is either provided directly in the input files within the Grid File System beforehand, or retrieved from third party sources during run time. As part of the forensics after discovery of the job, evidence may be found within the job's input files, showing either malicious data or a request to retrieve code from external sources. A user could delete or overwrite the files in the Grid File System to cover up traces. Nevertheless, with respect to the job's run time, the file alteration time stamp in the File Catalogue will record such changes. Since the File Catalogue has the property that physical files to which it refers are never overwritten or deleted directly, there exists a time window for recovering the original data of an affected file entry. Both delete and overwrite operations on the File Catalogue level lead to a simple disassociation of the affected physical files and shadow entries are stored in a central bookkeeping table [23]. The success of the recovery is limited by the time frame for deletion of stale files.

Conversely, a user would be able to disclaim responsibility for malicious code execution within a job when that job only referred to files uploaded to the Grid File System by other users. In that case the challenge is to find the evidence in those files.

In case of malicious software packages or an infected or attacked batch system or WN, a disclaimer of responsibility would be more difficult to assess. Nevertheless, the proposed mechanisms could ensure no counterfeit evidence for the job submitter's responsibility could be put in place.

If only storage systems are infected or attacked, the reference checksums in the File Catalogue will identify modified data, as the checksums are verified after the data has been received by the PJ or the job.

Finally, in case the File Catalogue is infected or attacked, it is possible to place counterfeit evidence claiming a job submitter's responsibility for a malicious job. This could be achieved due to file ownership, checksums, and modification timestamps being stored in the File Catalogue and the sJDL relying on these references. As such, the File Catalogue constitutes an entity which needs to be fully trusted by users and Sites. The same applies to the ticket service granting access to storage servers on the physical storage layer. If the PJ or the job receives illegimate tickets, not conforming to the specifications within the sJDL, accountability and verifiability are exposed.

4.4. Pilot platform integrity

Objective 8, the *Pilot platform integrity*, cannot be assured by the presented mechanisms and was introduced as an auxiliary criterion. Nevertheless, the presented approach of least-privilege Pilot Job credentials and Certified Grid jobs is able to largely simplify the assessment of illegal or improper behaviour in a Grid job's environment on a WN. By preventing Grid job submission with credentials stolen from a WN the mechanisms would impede the proliferation of attacks. Covering tracks through escalated privileges and masquerading would be impeded as well. Moreover, Grid user credentials cannot be compromised and user identities can be protected from misuse. Potentially exploited, malicious file uploads to the Grid File System can be detected and cleaned up, as the write privileges granted to a job are stated in its sJDL.

5. Related Work

Beyond the fulfillment of the defined security objectives 1-9, a basic functional concern and boundary condition was to identify approaches allowing the least invasive integration into the current architecture of the ALICE Grid Services, by implication not involving any additional remote callbacks or service invocations (see problem 6). Accordingly, for example GridShib-based [24] implementations or dynamic restricted delegation [25], both based on callback mechanisms, were disqualified. In Ref. [20] the signature of job requests is suggested as an approach to avoid a potential "mixup" of GuPCs within a VO.

Snelling et al. [26] proposed a model called *Explicit Trust Delegation* (ETD) to digitally sign job requests in the UNICORE Grid framework allowing for static delegation. In comparison to our work, ETD uses only one signature, either by the user or a trusted Grid portal, which in the latter case is consequently based on unrestricted delegation to the portal. Further, ETD does not distinguish entities such as broker and agent within the Grid framework, and gives no explicit information on intermediate processing, validation or the delegation's consequences for accountability on the execution endpoint.

6. Conclusion

This document examines security aspects of multi-user Grid jobs and underlines, while referring to potential implementations, crucial deficiencies of unrestricted delegation based on X.509 proxy credentials. Inspired by static delegation with warrant, a new model of mediated definite delegation is presented, allowing for dynamical assignment of definite privileges of delegators to agents by task delegations. Its implementation, based on compound digital signatures, establishes verifiable task and privilege delegation statements, thereby allowing for strong accountability and long-term traceability of Grid job submissions and establishing the foundation necessary for non-repudiation. The presented prototype further impedes the escalation of privileges and prevents identity theft on a WN. Its design foresees a potential interaction with the gLExec Grid middleware and constitutes a necessary framework to achieve full on-site user accounting and protection of Grid jobs and their environment.

7. Future Work

The presented prototype implementation of Certified Grid Jobs is part of a major revision of the AliEn Grid middleware, which will be further developed and tested within the next months, featuring a completely new security architecture. Within that process several aspects of the current discussion will be followed up. In particular: the formalization of the derivatives or transformation rules for the job requests; the application of access permissions on the physical storage level according to the access granted on the Grid File System; further improvement of file integrity assurance. We aim to incorporate in our delegation model the aspect of Grid file access both on the logical and on the physical layer and to provide for explicit file integrity assurance.
Finally, we will analyse and research the scope of security and protection of the Grid job environment on a WN (objective 8).

8. Acknowledgment

We would like to thank the gLExec development group for their valuable input and suggestions and their positive feedback. In addition, we would like to thank Olga Vladimirovna Datskova and Arsen Hayrapetyan for proofreading and their efforts, help and suggestions regarding the formal presentation and illustration of this work.

References

[1] ALICE Collaboration 2005 ALICE, Technical Design Report, Computing Tech. Rep. 92-9083-247-9 CERN/LHCC-2005-018

[2] ALICE Collaboration URL `http://aliceinfo.cern.ch/Collaboration/`

[3] AliEn URL `http://alien2.cern.ch/`

[4] Bagnasco S, Betev L, Buncic P, Carminati F, Cirstoiu C, Grigoras C, Hayrapetyan A A, Harutyunyan A, Peter A J and Saiz P 2008 AliEn: ALICE environment on the GRID *Journal of Physics: Conference Series* vol 119 part 6 p 062012

[5] Worldwide LHC Computing Grid (WLCG) URL `http://lcg.web.cern.ch/lcg/`

[6] AliRoot URL `http://aliceinfo.cern.ch/Offline/`

[7] Cooper D, Santesson S, Farrell S, Boeyen S, Housley R and Polk W 2008 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile RFC 5280 (Proposed Standard)

[8] Foster I and Kesselman C 1999 Globus: A Toolkit-Based Grid Architecture *The Grid: Blueprint for a New Computing Infrastructure* ed Foster I and Kesselman C (Morgan Kaufmann) pp 259–278

[9] Neuman B C 1993 Proxy-Based Authorization and Accounting for Distributed Systems *Proceedings of the 13th International Conference on Distributed Computing Systems* pp 283–291

[10] Tuecke S, Welch V, Engert D, Pearlman L and Thompson M 2004 Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile RFC 3820 (Proposed Standard)

[11] Kouril D and Basney J 2005 A Credential Renewal Service for Long-Running Jobs *Proceedings of the 6th IEEE/ACM International Workshop on Grid Computing* GRID '05 pp 63–68

[12] Basney J, Humphrey M and Welch V 2005 The MyProxy online credential repository *Software: Practice and Experience* vol 35, issue 9 pp 801–816

[13] Groep D, Koeroo O and Venekamp G 2008 gLExec: gluing grid computing to the Unix world *Journal of Physics: Conference Series* vol 119 part 6 p 062032

[14] Welch V, Foster I, Kesselman C, Mulmo O, Pearlman L, Gawor J, Meder S and Siebenlist F 2004 X.509 proxy certificates for dynamic delegation *Proceedings of the 3rd Annual PKI R&D Workshop*

[15] Ceccanti A 2007 A VOMS overview Tech. Rep. EGEE-II INFSO-RI-031688 NRENS and Grids Workshop Malaga URL `http://www.terena.org/activities/nrens-n-grids/workshop-06/slides/ceccanti-voms-overview.pdf`

[16] Joint Security Policy Group 2007 Grid security policy Tech. Rep. CERN-EDMS-428008 LCG EGEE Joint Security Policy Group URL `https://edms.cern.ch/file/428008/5/Security_Policy_V5.7a.pdf`

[17] Caballero J, Hover J, Litmaath M, Maeno T, Nilsson P, Potekhin M, Wenaus T and Zhao X 2010 gLExec and MyProxy integration in the ATLAS/OSG PanDA workload management system *Journal of Physics: Conference Series* vol 219 part 7 p 072028

[18] Caballero J, Maeno T, Nilsson P, Stewart G, Potekhin M and Wenaus T 2011 Improving Security in the ATLAS PanDA System Tech. Rep. ATL-SOFT-PROC-2011-025 CERN Geneva URL `http://cdsweb.cern.ch/record/1322206/files/ATL-SOFT-PROC-2011-025.pdf`

[19] Paterson S K 2008 LHCb gLExec Testing *CERN GDB Meeting Geneva* URL `http://indico.cern.ch/getFile.py/access?contribId=6&sessionId=4&resId=0&materialId=slides&confId=20235`

[20] Groep D 2006 glexec deployment models - local credentials and grid identity mapping in the presence of complex schedulers Tech. Rep. INFSO-RI-508833 Joint OSG EGEE Operations Workshop CERN Geneva URL `http://indico.cern.ch/materialDisplay.py?contribId=s4t2&sessionId=s4&materialId=0&confId=a062031`

[21] Grigoras C, Betev L, Saiz P and Schreiner S 2010 Optimization of Grid Resources Utilization: QoS-aware client to storage connection in AliEn *13th International Workshop on Advanced Computing and Analysis Techniques in Physics Research* URL `http://pos.sissa.it/archive/conferences/093/032/ACAT2010_032.pdf`

[22] BouncyCastleProvider URL `http://www.bouncycastle.org/`

[23] Schreiner S, Bagnasco S, Banerjee S S, Betev L, Carminati F, Datskova O V, Furano F, Grigoras A, Grigoras C, Lorenzo P M, Peters A J, Saiz P and Zhu J 2011 Securing the AliEn File Catalogue - Enforcing authorization with accountable file operations *To appear in: Proceedings of the 18th International Conference on Computing in High Energy and Nuclear Physics* CHEP 2010

[24] Barton T, Basney J, Freeman T, Scavo T, Siebenlist F, Welch V, Ananthakrishnan R, Baker B, Goode M and Keahey K 2006 Identity Federation and Attribute-based Authorization through the Globus Toolkit, Shibboleth, Gridshib, and MyProxy *Proceedings of the 5th Annual PKI R&D Workshop*

[25] Ahsant M, Basney J, Mulmo O, Lee A and Johnsson L 2006 Toward an On-Demand Restricted Delegation Mechanism for Grids *Proceedings of the 7th IEEE/ACM International Conference on Grid Computing* GRID '06 pp 152–159

[26] Snelling D F, van den Berghe S and Quian Li V 2004 Explicit trust delegation: Security for dynamic grids *Fujitsu Scientific and Technical Journal* vol 40 no 2 pp 282–294